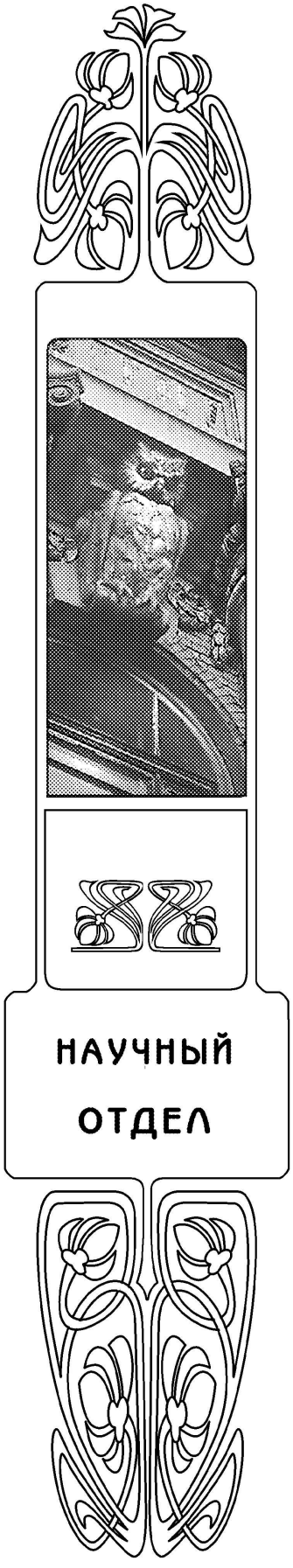
I. I. Slepovichev, P. V. Irmatov, M. S. Komarova, A. A. Bezhin, DDoS attack detection using fuzzy neural network,
Izv. Saratov Univ. Math. Mech. Inform., 2009, Volume 9, Issue 3, 84–89

https://www.mathnet.ru/eng/isu67



ИНФОРМАТИКА

УДК 519.71

ОБНАРУЖЕНИЕ DDoS АТАК НЕЧЕТКОЙ НЕЙРОННОЙ СЕТЬЮ

И.И. Слеповичев, П.В. Ирматов*, М.С. Комарова*, А.А. Бежин*

Саратовский государственный университет,
кафедра теоретических основ компьютерной безопасности и криптографии;
*Поволжский региональный центр новых информационных технологий,
отдел информационных ресурсов и систем
E-mail: gurgutan@yandex.ru, *karteron@yahoo.com

В статье анализируется DDoS атака SYN Flood. Предлагается метод для обнаружения атаки этого типа при помощи нечеткой нейронной сети. Приводится модификация алгоритма обратного распространения ошибки, используемая для обучения нейронной сети.

Ключевые слова: DDoS, SYN Flood, нечеткая нейронная сеть, алгоритм обратного распространения.

DDoS Attack Detection Using Fuzzy Neural Network

I.I. Slepovichev, P.V. Irmatov*, M.S. Komarova*, A.A. Bezhin*

Saratov State University,
Chair of Theoretical Basics of Computer Security and Cryptography;
* Volga Regional Center of New Informational Technologies,
Department of Informational Resources and Systems
E-mail: gurgutan@yandex.ru, *karteron@yahoo.com

In this article we analyze SYN Flood type of DDoS attack. We suggest method to detect this attack using a fuzzy neural network. Also we introduce modified back-propagation algorithm for neural network teaching.

Key words: DDoS, SYN Flood, fuzzy neural network, back-propagation algorithm

ВВЕДЕНИЕ

В наши дни распределенные атаки типа отказ в обслуживании (Distributed denial-of-service — DDoS) стали мощным инструментом в руках злоумышленников. Теперь это не просто способ тестирования сети при экстремальных нагрузках, но и средство шантажа владельцев интернет-сайтов. Большая часть таких атак — это атаки, нацеленные на вывод сервера из строя (то есть на отказ в обслуживании) путем посылки большого количества запросов на сервер. В результате канал связи переполняется, и компьютер жертвы не может отвечать на запросы пользователей. Это наносит значительный ущерб владельцам атакованных сетевых служб. На сегодняшний день разработаны методики противодействия DDoS атакам, однако их применение требует значительных материальных и административных затрат [1]. Обнаружение таких атак является непростой алгоритмической задачей, так как не существует простых и универсальных признаков, по которым можно было бы отличить сетевые запросы законопослушных пользователей к ресурсам сервера от запросов, посылаемых на сервер с целью атаки. В данной статье рассматривается абстрактная математическая модель DDoS атаки типа SYN Flood и предлагается способ её обнаружения на ранней стадии с использованием математического аппарата нечетких нейронных

сетей. Стоит отметить, что в данной работе не ставится задача дать универсальный способ обнаружения атак типа отказ в обслуживании, однако на примере этой прикладной задачи демонстрируются возможности математического аппарата нечетких нейронных сетей. Авторами данной статьи была создана программа, анализирующая входящий интернет-трафик и графически отображающая степень уверенности в наличии DDoS атаки.

1. АТАКА SYN FLOOD

Рассмотрим атаку SYN Flood. Целью такой атаки является вывод из строя сетевой службы так, чтобы атакуемый сервер не мог отвечать на запросы «обычных» пользователей. Для этого злоумышленником на адрес жертвы посылается большое количество пакетов данных переполняющих канал жертвы. Точнее, атака ведется на операционную систему с целью переполнения очереди поступающих пакетов. Так как размер очереди пакетов ограничен некоторой величиной N, а на обработку каждого запроса уходит определенное время T, то момент отказа в обслуживании пользователей будет определяться неравенством $v \geq N/T$, где v — скорость приходящих пакетов, то есть количество пакетов в секунду, N — длина очереди входящих соединений, T — время ожидания завершения соединения в секундах.

Значительного ущерба можно избежать, обнаружив атаку на ранней стадии. В этом случае у сетевого администратора будет время на принятие мер по предотвращению атаки или мер по минимизации возможного ущерба. Для решения этой задачи необходимо анализировать входящий трафик и если поступающая на сервер очередь пакетов данных имеет признаки начала атаки, сообщить об этом сетевому администратору.

Рассмотрим подробнее параметры очереди входящих пакетов данных, которые мы будем анализировать. Предположим, для анализа мы имеем последовательность из N пакетов. Анализировать будем всю последовательность, а не каждый пакет в отдельности. Существенными для анализа являются такие параметры, как TCP флаги (SYN) (нас интересуют пакеты только с этими флагами), IP адреса отправителя и получателя, направление пакета (входящее или исходящее), время поступления пакета и порт получателя. Также можно учитывать корректность заголовка: поступление большого количества пакетов с некорректными заголовками является одним из признаков атаки. Таким образом, на основе экспериментальных данных, был составлен вектор параметров для анализа, состоящий из следующих компонент:

[Процент пакетов с различными внешними IP адресами, процент пакетов с различными внешними портами, среднее время поступления одного пакета, процент пакетов с некорректными заголовками].

Эксперименты проводились на компьютере с процессором Pentium IV с частотой 3 ГГц, 512 Mb оперативной памяти. Для проведения атак использовалась виртуальная машина VMware с запущенной на ней ОС LINUX Freespire. С помощью виртуальной машины создавалась локальная сеть и сниффер, установленный на хостовой операционной системе, фиксировал параметры трафика. В качестве сниффера использовалась Trial версия программы SoftPerfect Analizer [2].

2. ПОСТРОЕНИЕ НЕЧЕТКОГО КЛАССИФИКАТОРА

Для решения задачи анализа был использован аппарат нечеткой логики и нейронных сетей [3, 4]. Для формализации знаний экспертов о DDoS атаке мы создали пять лингвистических переменных, каждая из которых характеризует одну из компонент вектора параметров. Напомним, что лингвистической переменной называют объект вида (B, T, X, G, M), где B — имя лингвистической переменной, T — множество наименований нечетких переменных, составляющих лингвистическую переменную, G — синтаксическая процедура, позволяющая оперировать элементами множества T, M — семантическая процедура, позволяющая переводить значения лингвистической переменной в нечеткую переменную. Далее приведены лингвистические переменные, созданные на основе экспериментальных данных. Процедура G нам не понадобится, поэтому мы пропускаем её в описании переменных.

1. (Время_поступления_пакетов, {«мало», «средне», «велико»}, [0,1000], $\{f_{11}(x) = \frac{1}{1+e^{-0.3(x-111)}}$, $f_{12}(x) = 1 - \frac{1}{1+e^{-0.3(x+15)}}$, $f_{13}(x) = e^{-(x-63)^2 2*0.025^2}\}$);

2. (Процент_различных_внешних_ip_адресов, {«мал», «велик»}, [0,100], $\{f_{21}(x) = \frac{1}{1+e^{-0.3(x-75)}}$, $f_{22}(x) = 1 - \frac{1}{1+e^{-0.3(x+25)}}\}$);

3. (Процент_различных_портов, {«мал», «велик»}, [0,100], $\{f_{31}(x) = \frac{1}{1+e^{-3(x-15)}}$, $f_{32}(x) = 1 - \frac{1}{1+e^{-2(x-3)}}\}$);

4. (Процент_пакетов_с_поврежденными_заголовками, {«велик»},[0,100], $\{f_{41}(x) = \frac{1}{1+e^{-2(x-5)}}\}$);

5. (Степень_уверенности, {«низкая», «средняя», «высокая»}, [0,100], $\{f_{51}(x) = \frac{1}{1+e^{-0.5(x-75)}}$, $f_{52}(x) = 1 - \frac{1}{1+e^{-0.5(x+25)}}$, $f_{53}(x) = e^{-(x-50)^2 2*15^2}\}$).

Нечеткий логический вывод для определения степени уверенности в атаке будет состоять из следующих предикатных правил:

1. Если Время_поступления_пакетов есть «мало» И Процент_различных_внешних_ip_адресов есть «мал» И Процент_различных_портов есть «мал», тогда Степень_уверенности_в_атаке есть «средняя»;

2. Если (Время_поступления_пакетов есть «мало» ИЛИ Время_поступления_пакетов есть «средне») И (Процент_различных_внешних_ip_адресов есть «велик» ИЛИ Процент_различных_портов есть «велик» ИЛИ Процент_пакетов_с_поврежденными_заголовками есть «велик»), тогда Степень_уверенности_в_атаке есть «высокая»;

3. Если Время_поступления_пакетов есть «велико», тогда Степень_уверенности_в_атаке есть «низкая»;

На основе нечеткого логического вывода был построен нечеткий классификатор со структурой, представленной на рис. 1.

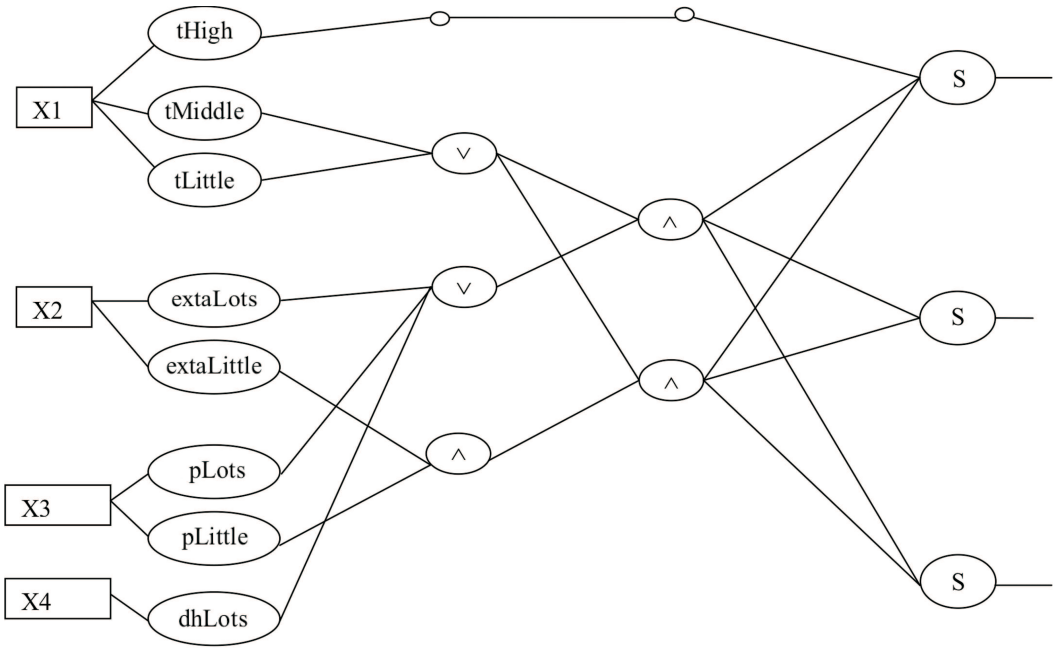


Рис. 1. Схема нечеткого классификатора для обнаружения SYN Flood атак

Здесь символом $\vee$ обозначен нечеткий ИЛИ-нейрон, символом $\wedge$ — нечеткий И-нейрон, символом S — классический нейрон, X1, X2, X3, X4 — соответствующие входы, а tLittle, tMiddle, tHigh, extraLittle, extraLots, pLittle, pLots, dhLots — функции активации для нечетких переменных Время_поступления_пакетов мало, Время_поступления_пакетов средне, Время_поступления_пакетов велико, Процент_различных_внешних_ip_адресов мал, Процент_различных_внешних_ip_адресов велик, Процент_различных_портов мал, Процент_различных_портов велик, Процент_пакетов_с_поврежденными_заголовками велик соответвтвенно.

Таким образом, построенный нечеткий классификатор должен по данным, поданным на вход, определять степень уверенности в атаке.

Данная нечеткая система может быть представлена в форме многослойной нейронной сети с прямым распространением сигнала. Указанное представление нечеткого классификатора дает нам возможность использовать такие достоинства нейронных сетей, как обучение и адаптация под конкрет-

ную ситуацию. Это может быть полезным в том случае, когда конкретные параметры лингвистических переменных неизвестны и подбирать их приходиться в процессе эксплуатации системы обнаружения атак. Нейронная сеть, реализующая нечеткий классификатор, будет иметь структуру, идентичную представленной на рис. 1. В этой сети присутствуют нейроны следующих видов: И-нейроны, вычисляющие значение функции нейчеткой конъюнкции, ИЛИ-нейроны, вычисляющие значение нечеткой дизъюнкции, нейроны, вычисляющие значения функций принадлежности нечетким множествам, и S-нейроны, вычисляющие выход нечеткого классификатора.

Для обучения этой нейронной сети мы можем использовать любой градиентный метод минимизации целевой функции. В нашей работе был использован метод обратного распространения ошибки.

3. ОБУЧЕНИЕ НЕЙРОННОЙ СЕТИ

Обучение нейронной сети происходит по методу обратного распространения ошибки, который состоит в следующем: на каждой итерации алгоритма обратного распространения весовые коэффициенты нейронной сети модифицируются так, чтобы улучшить решение одного примера. Таким образом, в процессе обучения циклически решаются однокритериальные задачи оптимизации [5].

Классический алгоритм для нашей нейронной сети не подходит, так как в нашей нейронной сети у различных нейронов различные функции активации, и нельзя записать общую формулу, которая бы подошла для всех нейронов сети, поэтому мы модифицировали классический алгоритм для наших целей.

Функция ошибки имеет следующий вид:

$$E = \frac{1}{2} \sum_{k \in Outputs} (t_k - o_k)^2,$$

где k — номер выхода нейронной сети пробегает все возможные выходы, t_k — желаемый выход, o_k — выход нейронной сети.

Тогда производная ошибки на последнем слое выражается формулой

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial S_j} \frac{\partial S_j}{\partial w_{ij}} = x_{ij} \frac{\partial E}{\partial S_j},$$

где w_{ij} — вес от нейрона i к нейрону j, $S_j = \sum\limits_{k \in inputs} w_{kj} x_k$.

Вычислим значения

$$\frac{\partial E}{\partial S_j} = \frac{\partial E}{\partial o_j} \frac{\partial o_j}{\partial S_j} = \left(\frac{\partial}{\partial o_j} \frac{1}{2} \sum_{k \in Outputs} (t_k - o_k)^2 \right) \left(\frac{\partial o_j}{\partial S_j} \right) = \frac{1}{2} \frac{\partial}{\partial o_j} (t_j - o_j)^2)(o_j(1-o_j) = -o_j(1-o_j)(t_j - o_j).$$

Теперь можно записать формулу подсчета ошибки для остальных слоев:

$$\frac{\partial E}{\partial w_{ij}} = \sum_{k \in Child} \frac{\partial E}{\partial S_k} \frac{\partial S_k}{\partial w_{ij}},$$

где S_k — это функция конкретного нечеткого нейрона.

Отсюда видно, что первая часть данной формулы, а именно $\frac{\partial E}{\partial S_k}$ — это ошибка на предыдущем шаге. Запишем вторую часть для каждого из слоев сети. Для предпоследнего слоя:

$$\frac{\partial S_k}{\partial w_{ij}} = \frac{\partial S_k}{\partial o_j} \frac{\partial o_j}{\partial w_{ij}}.$$

Для второго и третьего слоев, содержащих нечеткие нейроны получим

$$\frac{\partial S_k}{\partial w_{ij}} = \frac{\partial S_k}{\partial w_{ij}}. \tag{1}$$

Для И-нейронов под S_k будем понимать значение, вычисляемое по формуле

$$S_k = \prod_{k \in Inputs} (w_{kj} + x_{ki} - w_{ki} x_{ki}),$$

а для ИЛИ-нейронов — значение, вычисляемое по формуле

$$S_k = (w_{ij}x_{ij} + w_{i+1j}x_{i+1j} + w_{i+2j}x_{i+2j} + w_{ij}x_{ij}w_{i+1j}x_{i+1j} + \\ + w_{ij}x_{ij}w_{i+2j}x_{i+2j} - w_{ij}x_{ij}w_{i+1j}x_{i+1j}w_{i+2j}x_{i+2j}).$$

Тогда формула (1) для И-нейрона будет иметь вид

$$\frac{\partial S_k}{\partial w_{ij}} = (1 - x_i) \prod_{k \in Inputs = i} (w_{kj} + x_{kj} - w_{kj}x_{kj}). \tag{2}$$

Для ИЛИ-нейрона формула (1) в случае входного вектора с тремя компонентами, будет иметь вид

$$\frac{\partial S_k}{\partial w_{ij}} = (x_{ij} + w_{i+1j}x_{i+1j} + w_{i+2j}x_{i+2j} + x_{ij}w_{i+1j}x_{i+1j} + x_{ij}w_{i+2j}x_{i+2j} - x_{ij}w_{i+1j}x_{i+1j}w_{i+2j}x_{i+2j}). \tag{3}$$

Для первого слоя получим:

$$\frac{\partial f_1}{\partial a} = \frac{e^{-a(x-b)}(b-x)}{(1+e^{-a(x-b)})^2}, \qquad \frac{\partial f_1}{\partial b} = \frac{e^{-a(x-b)}a}{(1+e^{-a(x-b)})^2}, \tag{4}$$

$$\frac{\partial f_2}{\partial a} = -\frac{\partial f_1}{\partial a}, \qquad \frac{\partial f_2}{\partial b} = -\frac{\partial f_1}{\partial b}, \tag{5}$$

$$\frac{\partial f_3}{\partial a} = \frac{e^{-\frac{(x-a)^2}{2b^2}}(x-a)}{b^2}, \qquad \frac{\partial f_3}{\partial b} = \frac{e^{-\frac{(x-a)^2}{2b^2}}(x-a)^2}{b^3}, \tag{6}$$

где $\frac{\partial f_1}{\partial a}$, $\frac{\partial f_1}{\partial b}$, $\frac{\partial f_2}{\partial a}$, $\frac{\partial f_2}{\partial b}$, $\frac{\partial f_3}{\partial a}$, $\frac{\partial f_3}{\partial b}$ — производные функций активации на первом слое. Так как на первом слое нет весовых коэффициентов, и функция по x сложная, то эти производные имеют смысл поправки $\frac{\partial S_k}{\partial w_{ij}}$ в формуле (1).

Поскольку мы вычислили поправку для узлов последнего слоя и выразили поправку для узла более низкого уровня через поправки более высокого, то теперь мы можем посчитать поправку для каждого настраиваемого параметра сети. Имея формулу вычисления ошибки последнего слоя, можно вычислить ошибку предпоследнего и соответственно поправку параметров нейронов этого слоя. И так далее — поправка для нейронов слоя непосредственно вычисляется из значений входов и поправок нейронов следующего за ним слоя. Именно из-за этой особенности вычисления поправок алгоритм называется алгоритмом обратного распространения ошибки.

Таким образом, алгоритм обратного распространения ошибки для нашей нечеткой сети будет следующим.

1. Инициализация сети: весовым коэффициентам и смещениям сети присваиваются начальные значения.

2. Определение элемента обучающей выборки: (<текущий вход>, <желаемый выход>). Текущие входы $(x_0, x_1 \dots x_{N-1})$ должны различаться для всех элементов обучающей выборки.

3. Вычисление текущего выходного сигнала: текущий выходной сигнал определяется в соответствии с традиционной схемой функционирования нейронной сети.

4. Настройка синаптических весов.

Для настройки весовых коэффициентов используется рекурсивный алгоритм, который сначала применяется к выходным нейронам сети, а затем проходит сеть в обратном направлении до первого слоя. Синаптические веса настраиваются по формуле

$$w_{ij} = w_{ij} - \eta \frac{\partial E}{\partial w_{ij}},$$

где w_{ij} — вес от нейрона i или от элемента входного сигнала i к нейрону j, η — некоторая константа, задающая величину шага, $\frac{\partial E}{\partial w_{ij}}$ — производная функции ошибки по настраиваемому параметру.

Ошибка для последнего слоя:

$$\frac{\partial E}{\partial w_{ij}} = x_{ij}o_j(1 - o_j)(t_j - o_j).$$

Если нейрон с номером j принадлежит одному из слоев с первого по предпоследний, то

$$\frac{\partial E}{\partial w_{ij}} = \sum_{k \in Child} \frac{\partial E}{\partial S_k} \frac{\partial S_k}{\partial w_{ij}},$$

где $\frac{\partial E}{\partial S_k}$ — производная на предыдущем слое, а второй множитель — производная, подсчитанная по формуле для соответствующего нейрона сети. Например, для И-нейрона по формуле (2), а для ИЛИ-нейрона по формуле (3). Поправки для параметров функций принадлежности нечетким множествам, т. е. к первому слою, считаются по одной из соответствующих формул (4)–(6).

4. ПРОГРАММА

На основе вышеизложенной теории нами была создана программа. Программа состоит из:

– модуля классификатора, который может служить для проектирования искусственных нейронных сетей произвольной архитектуры и работы с ними;

– модуля сниффера пакетов, который перехватывает все пакеты, приходящие по протоколу TCP, и осуществляет их фильтрацию;

– модуля анализатора, который получает данные от сниффера пакетов, формирует из них векторы значений для анализа и передает нечеткому классификатору, который осуществляет оценку уровня опасности атаки.

При программировании модуля нечеткого классификатора были созданы классы для моделирования нечетких нейронных сетей, классы для обучения нечетких нейронных сетей по приведенному выше алгоритму.

Архитектура нейронной сети полностью задается через три конфигурационных файла. Два из них задают архитектуру нейронной сети, а третий задает коэффициенты для функций принадлежности нечетких множеств.

Обучение может происходить в двух режимах. В режиме реального времени пользователь выбирает уровень опасности, выступая в роли «учителя», моделирует атаку, и при нажатии кнопки «начать обучение» программа собирает обучающие выборки, с помощью которых обучается нейронная сеть. В режиме работы «из файла» пользователю предлагается выбрать файл, в котором сохранен образ атаки, что позволяет обучать программу не в момент моделирования атаки, а позже. Образы атаки можно подготовить с помощью самой программы. Для этого в программе есть специальные средства.

При тестировании программы нами были созданы ситуации с имитацией атаки на сервер. Программа с обученным классификатором показала хорошие возможности по обнаружению SYN Flood атаки.

ЗАКЛЮЧЕНИЕ

В ходе работы мы рассмотрели проблему определения DDoS атак типа SYN Flood нечеткими нейронными сетями. Был построен нечеткий классификатор, который на основании параметров входящего сетевого трафика может судить о степени уверенности в наличии DDoS атаки на компьютер. Также был предложен алгоритм обучения нейронной сети. В итоге была разработана программа, которая, используя математический аппарат нечеткой логики и нейронных сетей, определяет степень уверенности в наличии атаки.

Библиографический список

1. *Paxson V.* Bro: A System for Detecting Network Intruders in Real-Time // Computer Networks. 1999. V. 31. P. 2435–2463.

2. SoftPerfect Network Protocol Analyzer – Network sniffer for Windows // http://www.softperfect.com/products/networksniffer/

3. *Круглов В.В., Дли М.И., Голунов Р.Ю.* Нечеткая логика и искусственные нейронные сети. М., 2001. 224 с.

4. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы. М., 2006. 452 с.

5. *Rumelhart D.E., Hinton G.E., Williams R.J.* Learning internal representations by error propagation // Parallel Data Processing. 1986. V. 1. P. 318–362.